Helwan University

Computer and Artificial intelligence – Computer Sicnce

T42

Report about training Project

Anomaly Detetction for Satellite Telemetry Data Using Machine Learning

Prepared By

1- Yousef khaled elgioushi
2-Mohamed Ramzy gad
3- Mohamed maher fouad
4- Mustafa Mahmoud said
5- Yousef Medhat Galal
6- Mohamed Mustafa Anas

Under Supervision of

<< Eng. Ahmed Salama>>

September 2021

كل ما هو مكتوب باللون الأحمر مطلوب وضع البيانات فيه
حسب كل مجموعه

1 - Abstract

Project idea id detect of telemetary data tha come from satellites with using of machine learning algorithms, First Thing telemetry data is set of measurment and reading of embedded device Across time interval so we depends on Time Series analysis in this project ,we use two dataset consist of Date and Temprature records ,after preprocessing dataset we use time series and machine learning algorithm to detect future temp

2 - Project overview

This project aims to detect telemetry data that will come from satellites through varoius machine learning techniques and determine best algorithm accourding to mean squre error

3-Work Requirments

For our work, our implementation language is Python 3 in addition to several python and Anaconda libraries including Pandas library for data manipulation and processing, NumPy for array manipulation, Matplotlib for Data Visualization, Scikit-learn to calculate mean average error and mean square error between predicted and actual values of the validation subset of data and statsmodels library to use ARIMA model. Our work is done under Conda environment.

4 -Task

1-Data analysis

2-Data Preprocessing

3-Select Algorithms suitable for data

4-Predict data

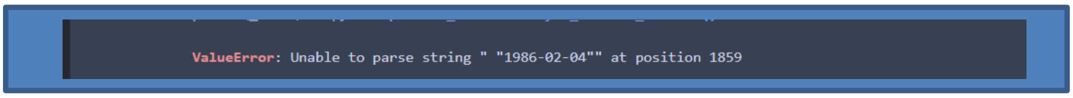
. 5-Compare between algorithms through mean square error

5 - Data analysis

First we load dataset 1 and 2 and try to figure them out, by see column , datatypes...etc

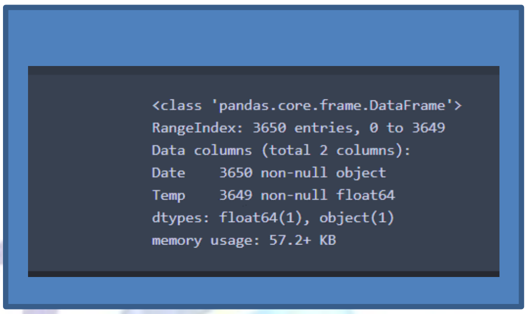
Then we found in **dataset 1 it** consists of one column "Date","Temp" and we try to seprate them into two columns 'Date' , 'temp' then convert temp column data type from string to float but we face an error:

Error:



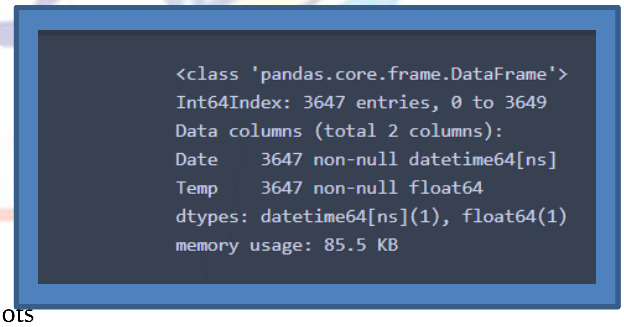
```
ValueError: Unable to parse string " "1986-02-04"" at position 1859
```

So we try to print the position 1859 and we found that data and temp are swapped with each other then ww fixed the error and covert temp data type to float



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3650 entries, 0 to 3649
Data columns (total 2 columns):
Date    3650 non-null object
Temp    3649 non-null float64
dtypes: float64(1), object(1)
memory usage: 57.2+ KB
```

Regarder to colum date we try to convert its data type To datetim data type so first we split date to Days , Months and years and check if there is error in Month And we found error in month with index 980 that month has value 33 so we drop this row from data set and we check about days as if days >31 but every thing was fine so we Concat day, month and year the genrate date and convert date from string to datetime and the drop Na and check for Duplicated but every thing was fine And dataset 1 Cleaned and ready for detection step in **dataset 2** it consists of one coulmn "Month","Sunspots" , as the previous dataset we try to seprate this column into two column Date and Sunspots and convert data type for Suspots from String to float and for Date w split it to month and year to check date month validation and we found two issue at index 7 where year and month are swapped ,at index 489 invalid month so we fix the first and drop the second then we concate between year and month and convert date from string to Datetime datatype the we drop rows that contain nan value and check for duplicated then the two data set was cleaned and ready .



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3647 entries, 0 to 3649
Data columns (total 2 columns):
Date    3647 non-null datetime64[ns]
Temp    3647 non-null float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 85.5 KB
```

6 - Algorithm Used:

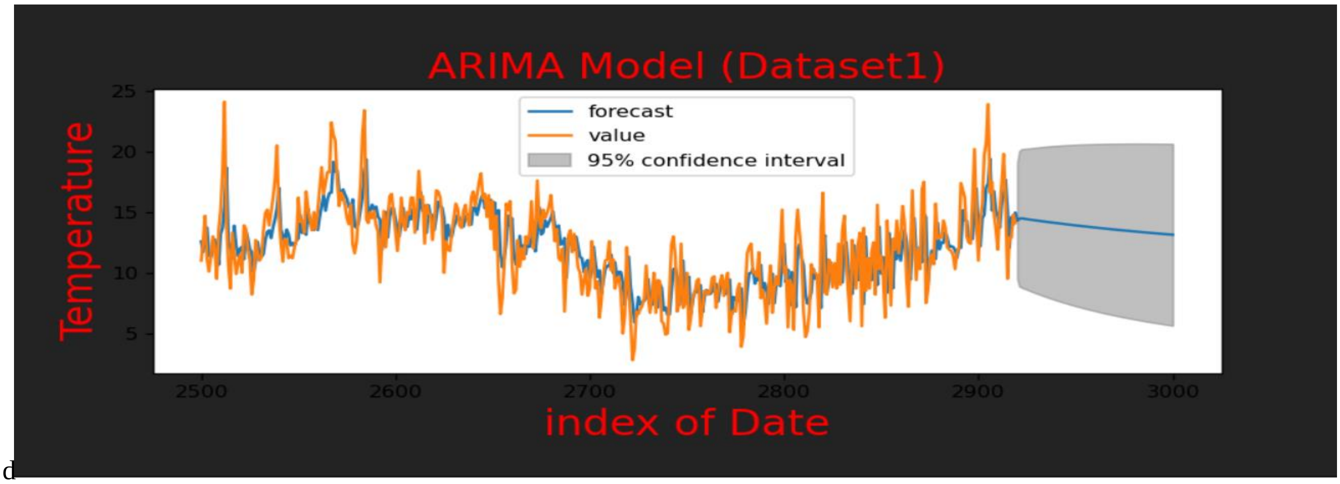
1-ARIMA

-brief description:

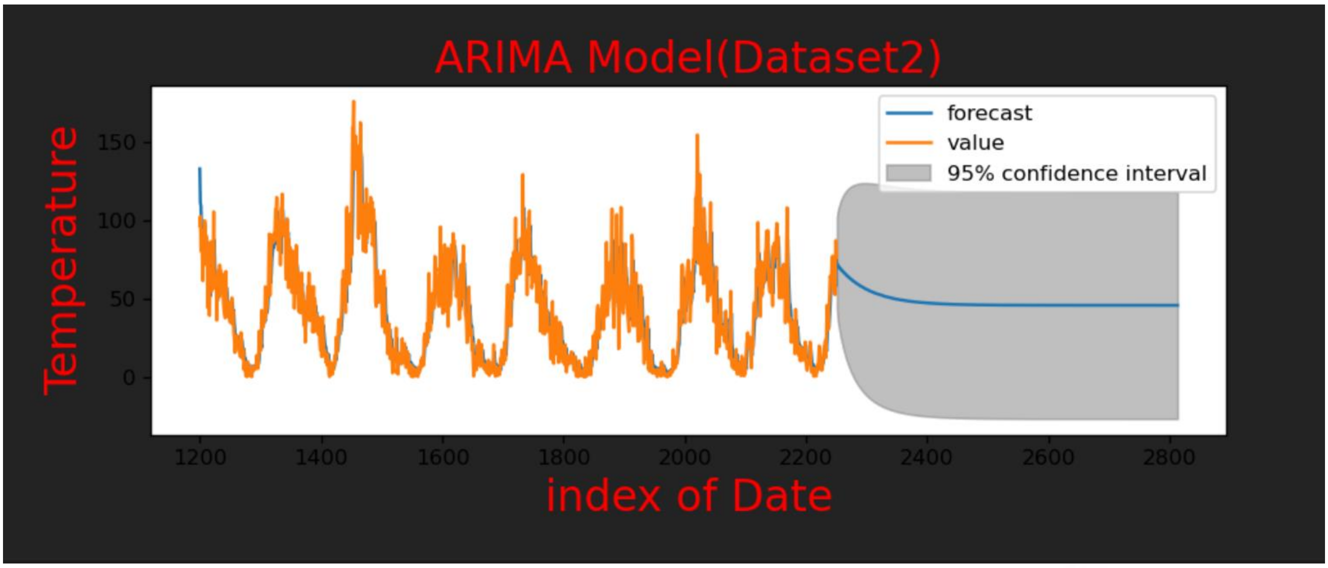
ARIMA, short for 'Auto Regressive Integrated Moving Average' is actually a class of models that 'explains' a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values.

- - how you implemented in project

1. First, we loaded our clean dataset 'Dataset_cleaned.csv' and shows its columns which were date and temp but we made it one column called "Value" to used easily in our model.
2. then we ran the model by importing 'statsmodels.tsa.arima_model' for model and import ndiffs to choose the best test model based on min value
3. create Auto ARIMA Model using pmdarima to get Best model base on lower AIC.
4. Fit our model.
5. And calculated mean squared error.
6. Finally, we plotted our graph.



d



-Result "Mean Square Error":

Test RMSE(Datset1): 4.01

Test RMSE(Datset2): 62.04

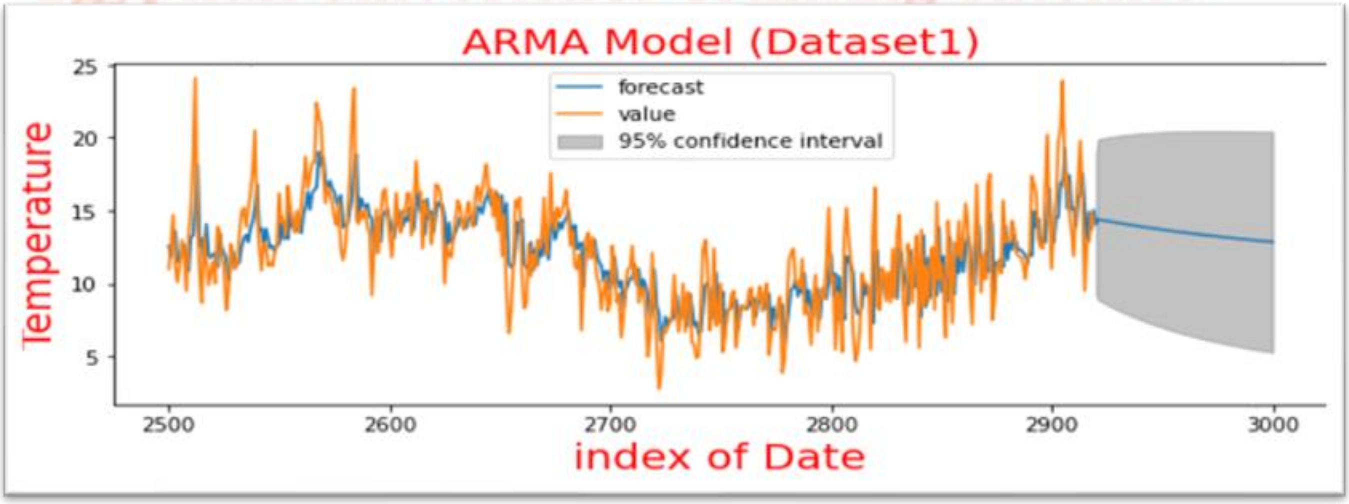
2-ARMA

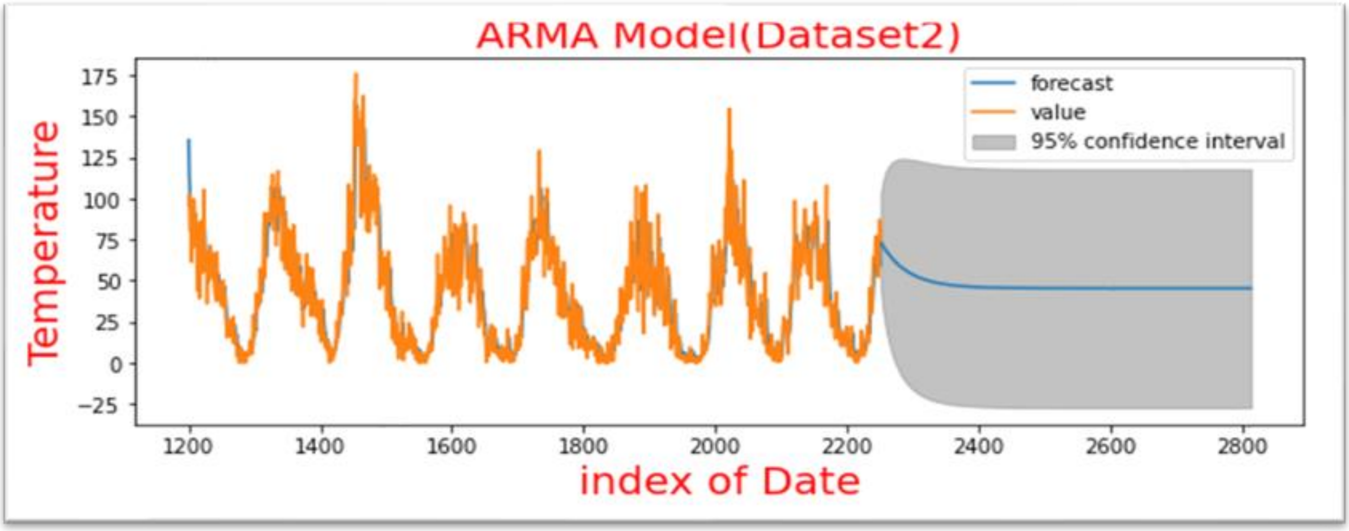
- **Brief Description**

  ARMA Model, or Autoregressive Moving Average Model, is used to describe weakly stationary stochastic time series in terms of two polynomials, often this model is referred to as the ARMA (p,d,q) model; where: p is the order of autoregressive polynomial, d is the order of the difference, q is the order of the moving average polynomial.

- **How you implemented in project**

  1. First, we loaded our clean dataset 'Dataset_cleaned.csv' and shows its columns which were date and temp but we made it one column called "Value" to use easily in our model.
  2. We plot the ACF and PACF to check the stationarity of data and if the plot is negatively correlated
  3. Since it wasn't stationary and negatively correlated we use differencing to make the data stationary
  4. After multiple tries to get the best results we will take second difference for the first dataset and first difference for the second dataset
  5. Create and Fit our model.
  6. we plotted our prediction graph.
  7. Finally, Calculated our mean square error





- Result “Mean Square Error”

Test RMSE(Datset1): 3.987

Test RMSE(Datset2): 62.247

3- Simple Exponential Smoothing "SES"

-brief discription:

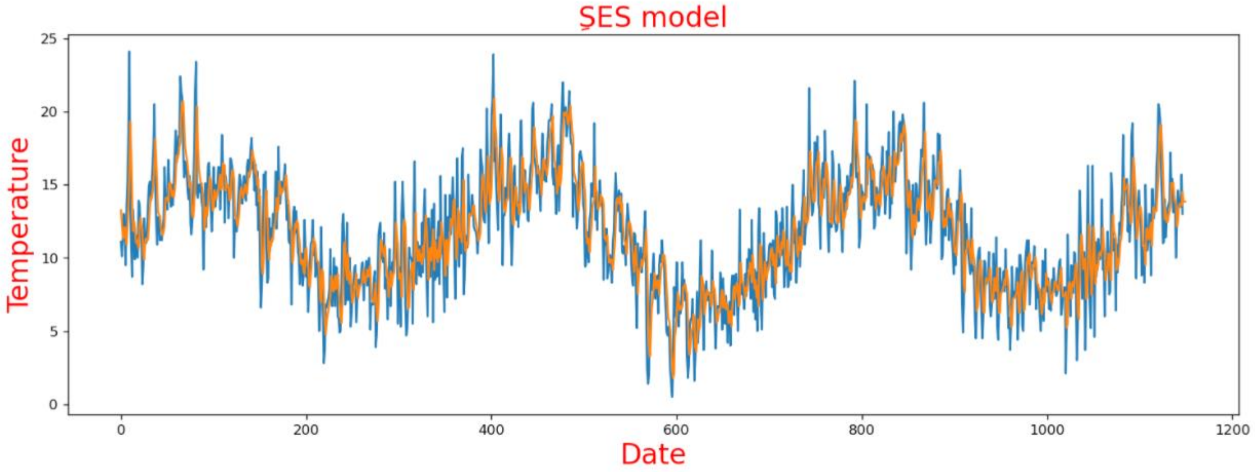
- Exponential Smoothing is an elementary and pragmatic technique used for forecasting where the forecast is made through the exponentially weighted average of prior observations.

- in its simplest form, an exponential smoothing of time series data allocates the exponentially decaying weights from newest to oldest observations, ie. analyzing data from a specific period of time via providing more importance to recent data and less importance to former data.

- how you implemented in project

For first Dataset:

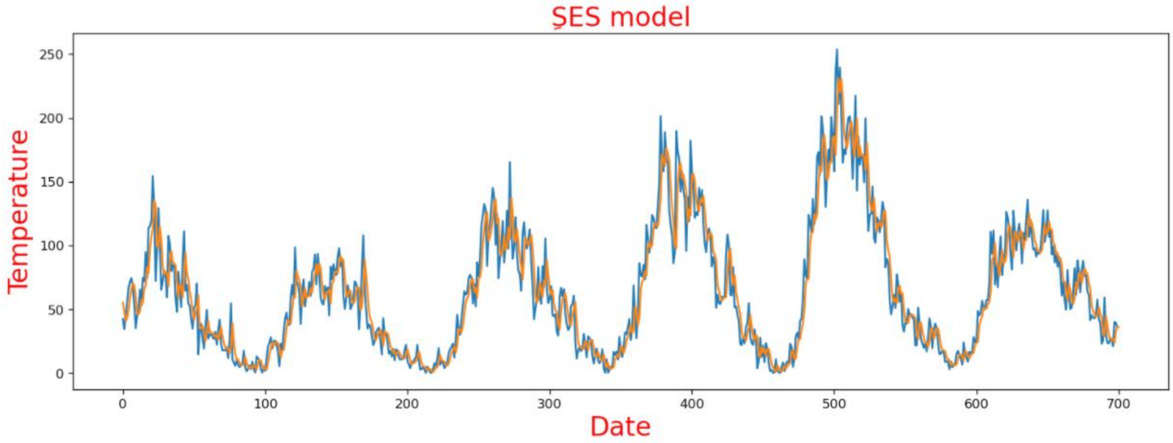
1. First, we loaded our clean dataset 'Dataset1_Cleaned.csv' and shows its columns which were date and temp but we made it one column called "Value" to used easily in our model.
2. then we ran the model by importing 'SimpleExpSmoothing' which is from statsmodels.tsa.holtwinters library
3. Defined our model.
4. Fit our model.
5. Then we made our prediction which ranged between `13.251537,12.082021,13.847238 and 14.506079`
6. The dimensions of the predicted data ((701,1)
7. And calculated mean squared error.
8. Finally, we plotted our graph.



Result of mean squared error ---->2.16876874275847

For second Dataset:

9. First, we loaded our clean dataset 'Datasets2_Cleanned.csv' and shows its columns which were date and temp but we made it one column called "Value" to used easily in our model.
10. then we ran the model by importing 'SimpleExpSmoothing' which is from statsmodels.tsa.holtwinters library
11. Defined our model.
12. Fit our model.
13. Then we made our prediction which ranged from `55.356835 to 36.115999`
14. The dimensions of the predicted data ((701,1)
15. And calculated mean squared error.
16. Finally, we plotted our graph.



-Result "Mean Square Error"

mean square error=17.21472624307182

4-Long Short Term Memory 'LSTM'

-brief discription

LSTMs are a type of Recurrent Neural Network (RNN) that can learn and memorize long-term

dependencies. Recalling past information for long periods is the default behavior.

- Long Short Term Memory (LSTM) is a type of deep learning model that is mostly used for analysis of sequential data (time series data prediction).
- There are different application areas that are used: Language model, neural machine translation, music generation, time series prediction, financial prediction, etc.
- The aim of this implementation is to help to learn structure of basic LSTM (LSTM cell forward, LSTM cell backward, etc..).

- how you implemented in project

1- load datasets

2-normalize data to be in range (0:1) using minmax scaler from sklearn.preprocessing

3- import keras from tenserflow

4-split data to training and testing sets

5- reshape data from 2d to 3d

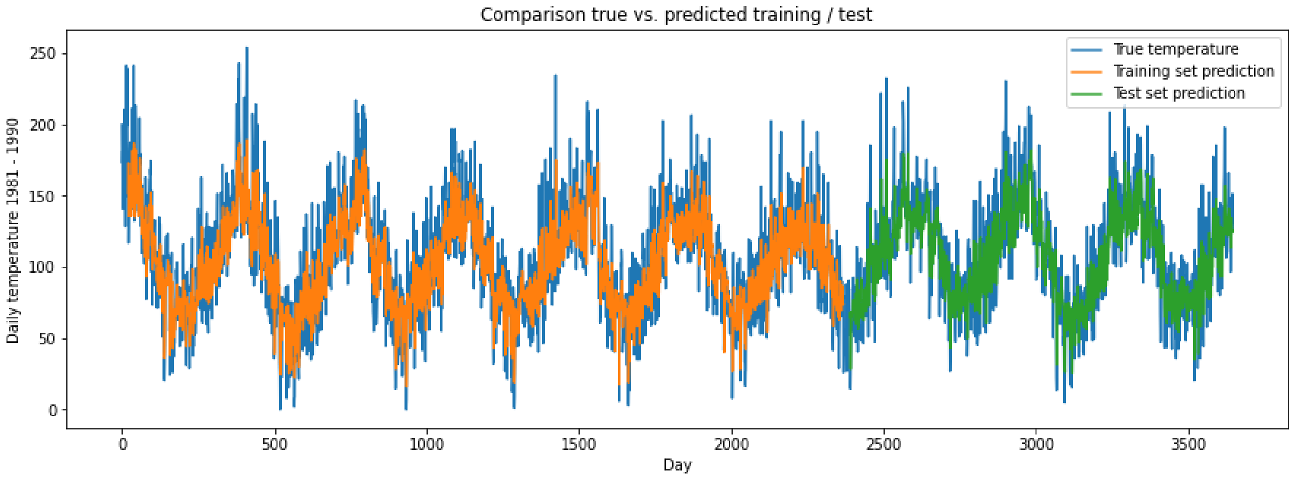
6-create LSTM sequential model using keras

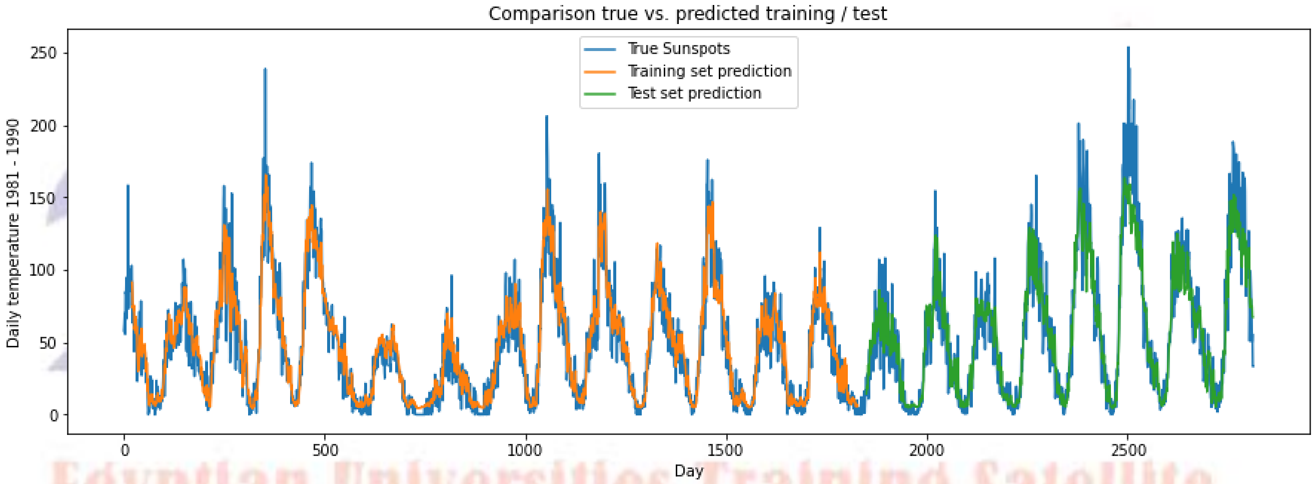
7- fit the model

8-plot loss & accuracy

9- plot training predictions , testing predictions and original data

10- calculate mean square error





-Result “Mean Square Error”

Training data 1 score: 23.87 RMSE

Test data 1 score: 23.15 RMSE

Training data 2 score: 13.86 RMSE

Test data 2 score: 18.49 RMSE

5-XGBosst Regression

-**brief description:**

- XGBoost is an efficient implementation of gradient boosting for classification and regression problems.

- XGBoost can also be used for time series forecasting, although it requires that the time series dataset be transformed into a supervised learning problem first. It also requires the use of a specialized technique for evaluating the model called walk-forward validation, as evaluating the model using k-fold cross validation would result in optimistically biased results.

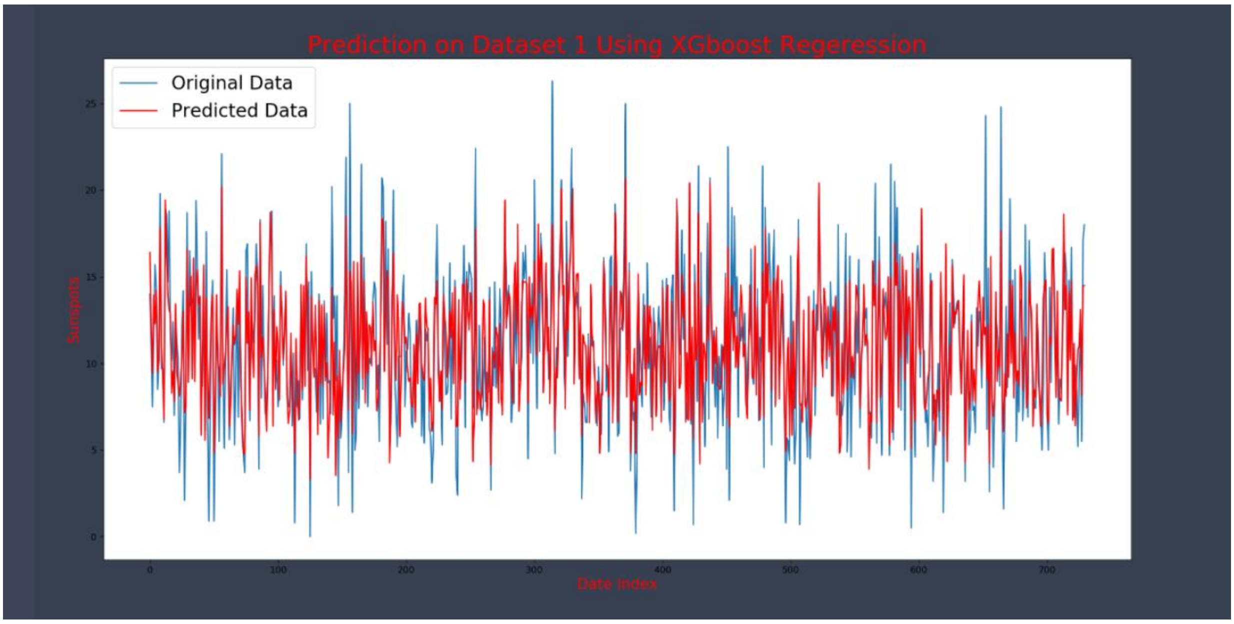
- how you implemented in project

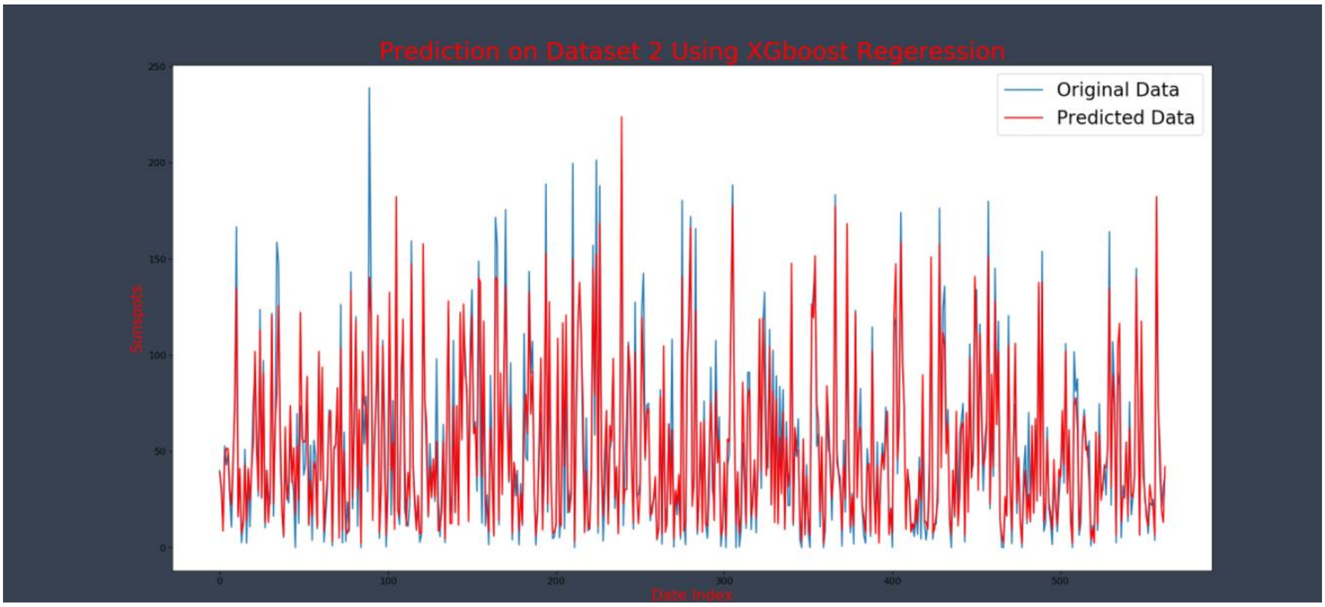
-First thig we Give Index Instead of Data to use XGBoost

-then, we split data to train and test with size 0.8 to train and without random state

- we try to find the best n_estimator through mean square error calculation then we select Best n_estimator for Dataset 1 is 100 and Dataset 2 is 70

-after that we predict our data and here is the prediction result on plot:





-Result Mean Square Error

For Dataset 1 mean Square error is 2091.96

Dataset 2 mean Square error is 1568.68

6- Gated recurrent Unit "GRU"

-brief discription

Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks, introduced in 2014 by Kyunghyun Cho et al. The GRU is like a long short-term memory (LSTM) with a forget gate, but has fewer parameters than LSTM, as it lacks an output gate.

GRUs are improved version of standard recurrent neural network to solve the vanishing gradient problem of a standard RNN, GRU uses, so-called, update gate and reset gate. Basically, these are two vectors which decide what information should be passed to the output.

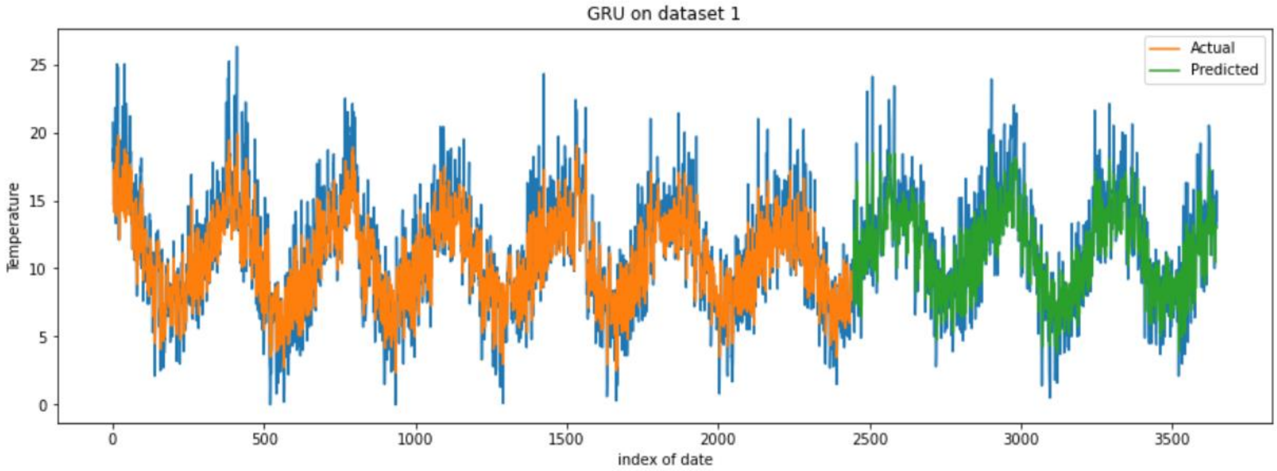
they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction.

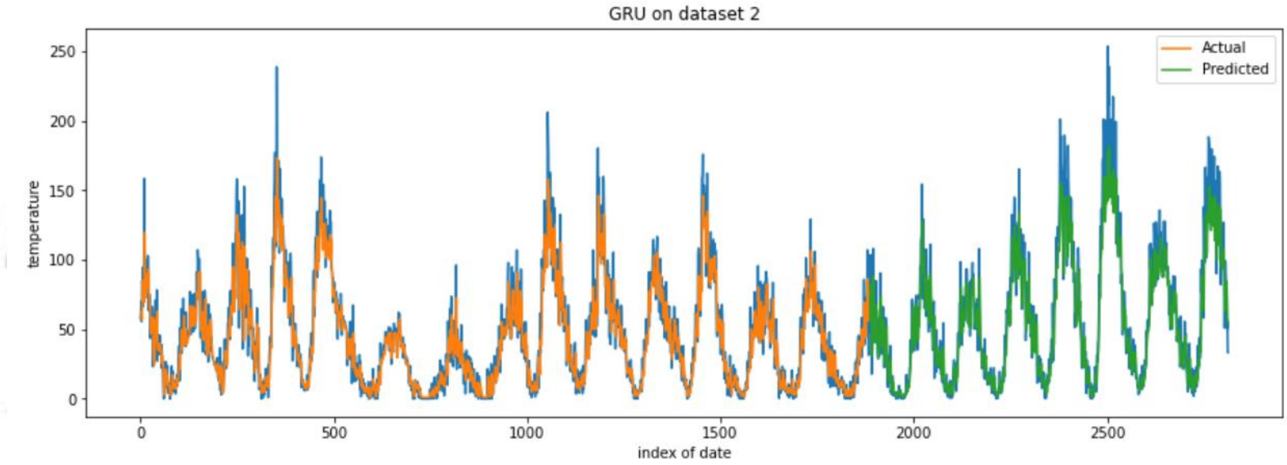
The update gate helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future.

The reset gate is used from the model to decide how much of the past information to forget.

- how you implemented in project

1. Load dataset
2. Fix random seed for reproducibility.
3. Normalize the dataset
4. Split the dataset into train and test sets with size 0.67
5. Convert an array of values into a dataset matrix
6. Reshape data x=t and y =t+1
7. Reshape input to be [samples, time steps features]
8. Create the GRU network and fit the model
9. Make predictions then invert them
10. Calculate root mean squared error
11. Plot data and predictions





-Result "Mean Square Error"

Dataset1

```
Train Score: 2.56 RMSE
Test Score: 2.47 RMSE
```

Dataset2

```
Train Score: 15.42 RMSE
Test Score: 18.84 RMSE
```

-Comparisom between algorithm performance:

Algorithm Name	Mean Square Error on Dataset 1	Mean Square Error on Dataset 2
ARIMA	4.01	62.04
ARMA	**3.987**	**62.247**
Simple Exponential Smoothing “SES”	**2.16**	**17.21**
Long Short Term Memory ‘LSTM’	23.15	18.49
XGBosst Regression	**1568.689**	**2091.9695**
Gated recurrent Unit “GRU”	2.56	15.42

7-Challenges

We Faced callnges as dataset was hard to be clean but we cleanned it at end and we was have no idea about time series analysis but after this project w gain a huge information about it

8- Mitigation Actions

Goals : Anomaly detection for temprature data that comes from statelite

Objectives: use Different algorithms as Arima, Arma , GRU , LSTm , XGBoost to predict temprature

Actions: we clean datatsets then try to prepare model of algorithms and implemet algorithm for prediction

9- Next step

Try To implemet model in Real Life